REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE November 30, 2006	3. REPORT TYPE AND DATES COVERED Final Report 4/28/06 – 11/30/06

4. TITLE AND SUBTITLE STRATA: Observer-based Measurement Agent Research Support Final Report	5. FUNDING NUMBERS N00014-06-M-0122
6. AUTHORS Jared Freeman, Frederick Diedrich, Jeanine Ayers	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Aptima, Inc. 12 Gill Street, Suite 1400 Woburn, MA 01801	8. PERFORMING ORGANIZATION REPORT NUMBER AP-R-1366
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Office of Naval Research One Liberty Center 875 North Randolph Street Arlington, VA 22203-1995	10. SPONSORING/MONITORING AGENCY Office of Naval Research
11. SUPPLEMENTARY NOTES	
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited. In accordance with the Small Business Administration Policy Directive (SBA PD) 2002, a public release decision will be executed by the parties in a written signed agreement after the contract award. This agreement must be appended to the report as front matter	12b. DISTRIBUTION CODE A

13. ABSTRACT *(Maximum 200 words)*

When learning to perform a task it is critical to have opportunities to practice the required behaviors. However, to be maximally effective, it is also the case that such practice needs to be accompanied by meaningful feedback to guide learning. Moreover, when considering how best to measure performance for feedback, it is critical to realize that there are multiple classes of performance measurement that are each essential, and unique, in their own right, including systems-based and observer-based measures. In the work reported here, we address this issue by focusing on a system that enables the collection and the integration of systems-based and observer-based measures. By providing an ability to collect and represent these measures in a common format, we have enabled the use of a wide range of feedback for trainees in simulation-based environments. Not only can complex measures be taken, but observer-based feedback can be integrated quickly and seamlessly, bypassing the traditional time associated with manual processing of observations.

14. SUBJECT TERMS Training, Feedback, Performance Measurement, Simulation-based Training, Observer-Based Measurement	15. NUMBER OF PAGES 50
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT

NSN 7540-01-280-5500 Computer Generated STANDARD FORM 298 (Rev 2-89)
Prescribed by ANSI Std 239-18
298-102

Table of Contents

Introduction

When learning to perform a task it is critical to have opportunities to practice the required behaviors. Hence, in order to effectively execute a certain type of mission, it is essential to be able to repeatedly attempt the needed skills, such as the correct communication patterns in air control, the correct formations to use when conducting a dismounted patrol, or the correct tactics to employ when conducting a Close Air Support mission. However, to be maximally effective, it is also the case that such practice needs to be accompanied by meaningful feedback to guide learning. Without such feedback, acquired skills may miss subtleties or at worst, may even diverge from standard protocols. *Systems that provide performance feedback are an essential accompaniment to environments for practice in order to create true training machines.*

Moreover, when considering how best to measure performance for feedback, it is critical to realize that there are multiple classes of performance measurement that are each essential, and unique, in their own right:

- *Systems-based Measures.* Systems-based measures allow trainers to measure aspects of performance from the simulation data stream that may be difficult to directly observe in real time. For instance, measures of weapons delivery within particular parameters (airspeed, dive angle, altitude) may be difficult for an observer to rapidly and accurately observe, making systems-based approaches ideal because the measurements can be pulled directly from a simulator data stream.
- *Observer-based Measures.* Observer-based measures are a critical complement to systems-based measures as they enable the assessment of items that difficult to directly record. As an example, accurate and reliable measures of communications and teamwork require observers, despite substantial advances in automated assessment. Furthermore, Observer-Controllers continue to be a key component of most military training.
- *Self-Report Measures*. Self-report measures as equally critical, for neither systems-based measures or observer-based measures alone can capture certain subjective, but equally essential, aspects of performance. Examples of these measures include things like workload or situation awareness that typically rely heavily on individual impressions.

Given, then, the importance of measurement as well as the different classes of measurement that are needed, there is a requirement for systems that can efficiently collect and present combined sources of information for feedback during training. In the work reported here, we address this issue by focusing on the integration of systems-based and observer-based measures. Although not addressed here given the scope of work reported, a similar strategy is being developed by Aptima for the integration of self-report measures into a cohesive performance measurement and feedback toolkit.

Measurement Objects for Human Performance in Simulations (MOHPS)

The basis of the work reported here is the development of flexible system for performance measurement, which Aptima is developing with NAVAIR and PMA-205 as part of the Measurement Objects for Human Performance in Simulations (MOHPS) project (e.g., Aptima, Inc., 2005). MOHPS is designed to provide a flexible, interoperable measurement infrastructure, and as such, it is being developed to provide an overall architecture that can ultimately encompass data capture, measurement, assessment, diagnosis, After Action Review, and configuration functions to map available performance measures to specific training instances. More specifically, the heart of the MOHPS project is the Performance Measurement Engine and the associated Performance Measurement Objects:

- The Performance Measurement (PM) Engine provides the capability to configure and collect performance measures from a simulator. It is the mechanism that connects to a simulator, gathers the requisite data, and computes the performance measures. The PM Engine handles the configuration, collection, and computation of performance measures in a stream-lined manner through direct use of Performance Measurement Objects (PMOs).
- PMOs are software objects in a federated HLA simulation. Their intent is to expand the world of HLA objects to include trainees, instructors, teams, assignments, performance measurements, and so on by describing human performance in a distributed virtual environment. Specifically, PMOs are used to communicate between the measurement engine and third-party simulator components, specify performance measurement configuration, and compute actual performance measures. PMOs are specified in Aptima's Human Performance Measurement Language (HPML). HPML is a platform-neutral medium for specifying PMOs that is universally machine-readable and easily decipherable (because it is XML-based). Importantly, HPML is not a programming language -- It is designed to express simple computations, referencing function libraries for more complex computations.

Collectively, MOHPS provides a method to develop and record performance measures that can be tied to a variety of simulations to ultimately provide feedback, and in particular, the MOHPS project is focused on developing the capability to handle systems-based measures.

Integration of Systems-based and Observer-based Measures

Building on this MOHPS capability, as part of follow-on work for the measurement portion of the Synthetic Teammates for Realtime Anywhere Training and Assessment (STRATA) effort of DARPA's DARWARS program, Aptima developed a basic capability to incorporate observer-based measurements. In particular, we created an Application Programming Interface (API) to link observer-based measures, collected on hand-held tablet computers, to the system-based measures. The observer assessment web service application at the heart of this work is a means to communicate observer based measurements to the performance measurement engine. As such, this work provides the basic capability to have both systems-based and observer-based measures in a common framework to allow computation when needed and display through eventual After Action Review tools. The API and Installation procedures have been previously delivered (Ayers & Diedrich, 2006; Kudzma, Ayers, & Diedrich, 2006). Below, we briefly describe accomplishments related to the current work for the S*TRATA: Observer-based Measurement Agent Research Support* project, which provided a small enhancement to the observer-based measurement integration work.

Summary of Technical Accomplishments

As part of the present effort, Aptima enhanced and demonstrated observer-based measurement capabilities to include time correlation with HLA events, generalization of question types, and further integration of automatic/observer measurement components. Related to this work, in Appendixes I and II we provide updated versions of the API and installation instructions. More specifically, our work accomplished the following items:

- *Correlation of timestamps of observer measures with the timestamps of HLA events.* Ultimately, the true power of integration of systems-based and observer-based measures depends on using those measures in a manner that captures their relative contribution to various events. For instance, knowing that a weapon was released within particular parameters might be especially powerful if it is also known that a particular communication occurred at the same time. As part

of this work, therefore, we created the ability to correlate observer measures with timestamps of HLA events by marking the time of observations and storing them in a database. This capability enables the presentation of feedback from multiple sources over time and with respect to mission stage during After Action Review.

- *Ability to compute measures that have both automatic and observer-based components.* In addition to the capability to correlate measures across time, it may also be advantageous to be able to calculate measures that have both automatic and observer-based components. Continuing with the previous example, for instance, it might be advantageous to construct a higher-order measure that captures performance related to both the communication and the weapons release. Such a measure could provide feedback on total weapons release behavior, which could then be dissected into components that are in this case both observer- and systems-based. Thus, the underlying performance measurement architecture was expanded to manage the connection to and collection from multiple data sources. In the past, only one source could be referenced at a time. The architecture has been expanded to allow multiple sources of data to contribute to the overall calculation of a measure, thus enabling eventual calculation and/or assessments of measures across measurement types.
- *Generalization of questions and probes from observer-based tools.* Initial work on integration of observer measures supported relatively few capability measure types (i.e., item response formats). In the work reported here, these measure types were expanded to handle different formats (rating, yes/no, or written observation). Currently, the API allows you to update and add questions, probes, and results from observer-based tools. The system allows the user to specify scale for the measurement, as well as measurement type. In total this capability provides a method to handle a variety of different measurement types to meet the needs of observers.
- *Demonstration of capability.* Based on this work, we demonstrated the capability of integration of observer-based measures with the MOHPS measurement engine in July 2006 to NAVAIR (Danielle Merket). In this case, we employed Aptima's SPOTLITE observer-based measurement tool to demonstrate that an observer measure collected on a tablet PC could be conveyed to the MOHPS measurement engine. In addition, in December 2007, this capability will once again be demonstrated at Pax River to the MH-60R community, using Aptima's SPOTLITE tool with sample observer measures developed as part of the VIRTE program (Wiese, Freeman, & Jackson, 2006). These measures will be taken along with system-based measures for the MH-60R, thereby showing the ability to capture various sources of data on performance. Note that although these demonstrations involved Aptima's SPOTLITE observer-based tools, other observer-based tools could be integrated using the API developed here (Appendix I).

Conclusions

In total, the work reported here represents a substantial step forward for performance measurement and assessment for training. By providing an ability to collect and represent both systems-based and observer-based measures in a common format, we have enabled the use of a wide range of feedback for trainees in simulation-based environments. Not only can complex measures be taken, but moreover, observer-based feedback can be integrated quickly and seamlessly, bypassing the traditional time associated with manual processing of observations. In future work, we plan to integrate the ability to capture self-report measures, thereby providing a common framework for a variety of complementary assessment techniques. In addition, we plan to develop methods to effectively create measurement plans across measurement types in order to plan feedback and to capitalize on these various measures during After Action Reviews. Collectively, these combined methods of measurement will enable feedback that is comprehensive and precise, thereby maximally facilitating training effectiveness.

References

Aptima, Inc. (2005). *Measurement objects for human performance in simulations (MOHPS): Year I summary report*. Aptima Technical Report, Aptima, Inc., Woburn, MA.

Ayers, J., & Diedrich, F. (2006). *Observer Assessment Web Service Application Programming Interface.* Aptima Technical Report, Aptima, Inc., Woburn, MA.

Kudzma, K., Ayers, J., & Diedrich F. (2006). *Installation Manual*. Aptima Technical Report, Aptima, Inc., Woburn, MA.

Wiese, E., Freeman, J., & Jackson, C. (2006). *VIRTE: DRAFT MH-60R Candidate Performance Measures*. Aptima Technical Report Number, Aptima, Inc., Woburn, MA.

Appendix I

Observer Assessment Web Service Application Programming Interface

The purpose of this document is to provide specification for an Application Programming Interface (API) to link observer-based measures, collected on hand-held tablet computers, to a system that captures system-based measures. As described here, the observer assessment web service application is a means to communicate observer based measurements to Aptima's performance measurement engine. More specifically, Aptima has developed the capability to link observer-based measures into its Measurement Objects for Human Performance in Simulations (MOHPS) framework, developed previously with NAVAIR and PMA-205. MOHPS provides an overall architecture that can ultimately encompasses data capture, measurement, assessment, diagnosis, During Action Review, After Action Review, and configuration functions to map available performance measures to specific training instances through Performance Measurement Objects (PMOs).

Below, we describe the process through which measures from hand-held tools can be communicated to the performance measurement system. The system was developed to enable connection of Aptima's SPOTLITE observer measurement tool. However, the API below provides the ability for other tools to be easily integrated. As described below, the basic steps are:

- Insert an observer assessment period that represents a unique assessment time period with a start and end date
- Insert a trainee profile, if available, to attribute measurement instances against
- Insert measures and measurement instances within the unique assessment period

Questions regarding use of this API should be addressed to Aptima via Jeanine Ayers or Emily Wiese, Aptima, Inc., 12 Gill St., Suite 1400, Woburn, MA 01801, 781-935-3966, jayers@aptima.com, ewiese@aptima.com.

ServiceObserverMeasure

PMO Engine - Observer Measure Service

The following operations are supported. For a formal definition, please review the **Service Description**.

- **InsertObserverAssessmentPeriod**
  Create a new **observer assessment period and attributes** for a specific starting date.
- **InsertObserverAssessmentPeriodWithEndDate**
  Create a new **observer assessment period and attributes** for a specific date range.
- **UpdateObserverAssessmentPeriodEndDate**
  Update an existing **observer assessment period end date**.
- **InsertPerformancePeriod**
  Create a new **performance period and attributes** for an observer assessment.
- **InsertTrainee**
  Create a new **trainee** for a specific observer assessment period.
- **InsertMeasurement**
  Create a new **measurement** for a specific question and measurement type.
- **InsertMeasurementInstance**
  Create a new **measurement instance and observations** for an observer assessment period.
- **UpdateMeasurementInstance**
  Update an existing **measurement instance and observations**.
- **RemoveMeasurementInstance**
  Remove an existing **measurement instance and observations**.

ServiceObserverMeasure

Click **here** for a complete list of operations.

InsertObserverAssessmentPeriod

Create a new **observer assessment period and attributes** for a specific starting date.

*ObserverResult **InsertObserverAssessmentPeriod** (moduleName, periodStartDate, atts, trainees)*

In:

Parameter	Type	Description
moduleName	string	This is the name of the module. Allowable types are A2A, Attack Coordinator, DTC Chief, Target Duty Officer, and Third Party.
periodStartDate	DateTime	This is a starting date and time of the observer assessment period.
atts	**ObserverAssessmentPeriodAttribute**[]	This is an array of attributes for the observer assessment period.
trainees	**Trainee**[]	This is an array of trainees.

Out:

Parameter	Type	Description
ObserverResult	**ObserverResult**	This is the result of the web method.

ServiceObserverMeasure

Click **here** for a complete list of operations.

InsertObserverAssessmentPeriodWithEndDate

Create a new **observer assessment period and attributes** for a specific date range.

ObserverResult
InsertObserverAssessmentPeriodWithEndDate
(moduleName, periodStartDate, periodEndDate, atts, trainees)

In:

Parameter	Type	Description
moduleName	string	This is the name of the module. Allowable types are A2A, Attack Coordinator, DTC Chief, Target Duty Officer, and Third Party.
periodStartDate	DateTime	This is a starting date and time of the observer assessment period.
periodEndDate	DateTime	This is an ending date and time of the observer assessment period.
atts	**ObserverAssessmentPeriodAttribute**[]	This is an array of attributes for the observer assessment period.
trainees	**Trainee**[]	This is an array of trainees.

Out:

Parameter	Type	Description
ObserverResult	**ObserverResult**	This is the result of the web method.

ServiceObserverMeasure

Click **here** for a complete list of operations.

UpdateObserverAssessmentPeriodEndDate

Update an existing **observer assessment period end date**.

*ObserverResult **UpdateObserverAssessmentPeriodEndDate** (observerAssessmentPeriodId, periodEndDate)*

In:

Parameter	Type	Description
observerAssessmentPeriodId	long	This is the unique identifer of an existing observer assessment period.
periodEndDate	DateTime	This is an ending date and time of the observer assessment period.

Out:

Parameter	Type	Description
ObserverResult	**ObserverResult**	This is the result of the web method.

ServiceObserverMeasure

Click **here** for a complete list of operations.

InsertPerformancePeriod

Create a new **performance period and attributes** for an observer assessment.

*ObserverResult **InsertPerformancePeriod** (observerAssessmentPeriodId, performancePeriodName, atts)*

In:

Parameter	Type	Description
observerAssessmentPeriodId	long	This is the unique identifer of an existing observer assessment period.
performancePeriodName	string	This is the name of the performance period.
atts	**PerformancePeriodAttribute**[]	This is an array of attributes for the performance period.

Out:

Parameter	Type	Description
ObserverResult	**ObserverResult**	This is the result of the web method.

ServiceObserverMeasure

Click **here** for a complete list of operations.

InsertTrainee

Create a new **trainer** for a specific observer assessment period.

*ObserverResult **InsertTrainee** (observerAssessmentPeriodId, job, name)*

In:

Parameter	Type	Description
observerAssessmentPeriodId	long	This is the unique identifer of an existing observer assessment period.
job	string	This is the trainee's job title.
name	string	This is the name of the trainee.

Out:

Parameter	Type	Description
ObserverResult	**ObserverResult**	This is the result of the web method.

ServiceObserverMeasure

Click **here** for a complete list of operations.

InsertMeasurement

Create a new **measurement** for a specific question and measurement type.

*ObserverResult **InsertMeasurement** (measurementQuestionText, measurementTypeName)*

In:

Parameter	Type	Description
measurementQuestionText	string	This is the question that is being measured.
measurementTypeName	string	This is the type of question that is being measured. Recommended types are LikertScale, YesNo, and MultipleOption.

Out:

Parameter	Type	Description
ObserverResult	**ObserverResult**	This is the result of the web method.

ServiceObserverMeasure

Click **here** for a complete list of operations.

InsertMeasurementInstance

Create a new **measurement instance and observations** for an observer assessment period.

*ObserverResult **InsertMeasurementInstance** (observerAssessmentPeriodId, measurementId, performancePeriodId, timestamp, obs)*

In:

Parameter	Type	Description
observerAssessmentPeriodId	long	This is the unique identifer of an existing observer assessment period.
measurementId	long	This is the unique identifer of an existing measurement.
performancePeriodId	long	This is the unique identifer of an existing performance period.
timestamp	DateTime	This is a time stamp of the measurement instance.
obs	**Observation** []	This is an array of observations.

Out:

Parameter	Type	Description
ObserverResult	**ObserverResult**	This is the result of the web method.

ServiceObserverMeasure

Click **here** for a complete list of operations.

UpdateMeasurementInstance

Update an existing **measurement instance and observations**.

*ObserverResult **UpdateMeasurementInstance** (measurementInstanceId, timestamp, obs)*

In:

Parameter	Type	Description
measurementInstanceId	long	This is the unique identifer of an existing measurement instance.
timestamp	DateTime	This is a time stamp of the measurement instance.
obs	**Observation** []	This is an array of observations.

Out:

Parameter	Type	Description
ObserverResult	**ObserverResult**	This is the result of the web method.

ServiceObserverMeasure

Click **here** for a complete list of operations.

RemoveMeasurementInstance

Remove an existing **measurement instance and observations**.

*ObserverResult **RemoveMeasurementInstance** (measurementInstanceId)*

In:

Parameter	Type	Description
measurementInstanceId	long	This is the unique identifer of an existing measurement instance.

Out:

Parameter	Type	Description
ObserverResult	**ObserverResult**	This is the result of the web method.

ServiceObserverMeasure

Click **here** for a complete list of operations.

Observation

Complex type for the observation.

Name	Type	Description
Value	string	This is the value of an observation.
DataType	string	This is the data type of the observation value. Recommended types are string, bool, int, long, and float.

ServiceObserverMeasure

Click **here** for a complete list of operations.

ObserverAssessmentPeriodAttribute

Complex type for the observer assessment period attribute.

Name	Type	Description
AttributeName	string	This is the name of an attribute.
AttributeValue	string	This is the value of an attribute.

ServiceObserverMeasure

Click **here** for a complete list of operations.

ObserverResult

Complex type for the result of the web method.

Name	Type	Description
StatusFlag	boolean	This is the status flag for the result of the web method. The flag is equivalent to true if the status is okay and false if it is not okay.
Value	object	This is a value for the result of the web method when the status flag is equivalent to true. This value is equivalent to null when a value is not a necessity.
Description	string	This is the description of the value when the value is not equivalent to null.
Message	string	This is an error message.

ServiceObserverMeasure

Click **here** for a complete list of operations.

ObserverResult

Complex type for the result of the web method.

Name	Type	Description
StatusFlag	boolean	This is the status flag for the result of the web method. The flag is equivalent to true if the status is okay and false if it is not okay.
Value	object	This is a value for the result of the web method when the status flag is equivalent to true. This value is equivalent to null when a value is not a necessity.
Description	string	This is the description of the value when the value is not equivalent to null.
Message	string	This is an error message.

ServiceObserverMeasure

Click **here** for a complete list of operations.

Trainee

Complex type for the trainee.

Name	Type	Description
Job	string	This is the trainee's job title.
Name	string	This is the name of the trainee.
ObserverAssessmentPeriodId	long	This is the unique identifer of an existing observer assessment period.

ServiceObserverMeasure

Click **here** for a complete list of operations.

PerformancePeriodAttribute

Complex type for the performance period attribute.

Name	Type	Description
AttributeName	string	This is the name of an attribute.
AttributeValue	string	This is the value of an attribute.

Example Program and Integration Steps

Requirements:

1. Java JDK 1.4 or greater (http://java.sun.com/)
2. Apache Axis version 1.2 (http://ws.apache.org/axis/)
3. Access to http://pmoengine.aptima.com

Instructions:

1. **Download wsdl file**
   a. Long on as any user to http://pmoengine.aptima.com
   b. Navigate to URL **http:// pmoengine.aptima.com /services/ObserverMeasure?wsdl**
   c. Select File->Save As (the default filename som.xml work fine) to save the wsdl to your local machine.

2. **Generate service stubs and Java classes**
   a. Add all jar files included in the Axis download (%AXIS_HOME%/lib) to your class path.
   b. Navigate to the location of the downloaded wsdl (som.xml) from a console window.
   c. Run the command: *java org.apache.axis.wsdl.WSDL2Java som.xml*
   This will create two directories, com and org, which will contain all of the code necessary to execute the Observer Assessment WebServices. **Note: other command switches for WSDL2Java can be added. Only the default usage is shown here. Consult http://ws.apache.org/axis/java/reference.html#WSDL2JavaReference for more information.**

3. **Using the WebServices**
   a. Create a new Java project using an IDE of your choice.
   b. Add the generated files from step 2 (com and org directories) into the project.
   c. Make sure the entire axis jars are added to the classpath.
   d. Begin using all the methods as defined by the Observer Assessment interface. These methods will be defined in the pmoengine.java generated file in package

Test Program

```
        //this method returns a ServiceObserverMeasure interface
                public ServiceObserverMeasureSoap getObserverMeasureService() {

                ServiceObserverMeasureLocator observerMeasureServiceLocator = new
ServiceObserverMeasureLocator();
                ServiceObserverMeasureSoap observerMeasureSoapBindingStub = null;

          try {
                observerMeasureSoapBindingStub =
                        observerMeasureServiceLocator.getServiceObserverMeasureSoap();
          }
          catch (Exception e1) {
           e1.printStackTrace();
          }
          return observerMeasureSoapBindingStub;
         }


       public NetworkSpotliteTest() {
      ObserverResult result = null;

          ServiceObserverMeasureSoap soapService = this.getObserverMeasureService();
          Calendar rightNow = Calendar.getInstance();

          ObserverAssessmentPeriodAttribute atts = new ObserverAssessmentPeriodAttribute();
          atts.setAttributeName("syllabus");
          atts.setAttributeValue("low slow flyer");
          ObserverAssessmentPeriodAttribute data[] = {atts};

          try {
                ObserverResult result =
(ObserverResult)soapService.insertObserverAssessmentPeriod("A2A",rightNow,aAttrs);
          }
          catch (Exception e2) {
                e2.printStackTrace();
          }
       }
```

WSDL

```
<wsdl:definitions targetNamespace="http://PMOEngine.aptima.com">
<wsdl:documentation><b>PMO Engine - Observer Measure Service</b></wsdl:documentation>
–
        <wsdl:types>
–
        <s:schema elementFormDefault="qualified" targetNamespace="http://PMOEngine.aptima.com">
–
        <s:element name="InsertObserverAssessmentPeriod">
–
        <s:complexType>
–
        <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="moduleName" type="s:string"/>
<s:element minOccurs="1" maxOccurs="1" name="periodStartDate" type="s:dateTime"/>
<s:element minOccurs="0" maxOccurs="1" name="atts" type="tns:ArrayOfObserverAssessmentPeriodAttribute"/>
<s:element minOccurs="0" maxOccurs="1" name="trainees" type="tns:ArrayOfTrainee"/>
</s:sequence>
</s:complexType>
</s:element>
–
        <s:complexType name="ArrayOfObserverAssessmentPeriodAttribute">
–
        <s:sequence>
<s:element minOccurs="0" maxOccurs="unbounded" name="ObserverAssessmentPeriodAttribute" nillable="true"
type="tns:ObserverAssessmentPeriodAttribute"/>
</s:sequence>
</s:complexType>
–
        <s:complexType name="ObserverAssessmentPeriodAttribute">
–
        <s:complexContent mixed="false">
–
        <s:extension base="tns:BusinessObject">
–
        <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="AttributeName" type="s:string"/>
<s:element minOccurs="0" maxOccurs="1" name="AttributeValue" type="s:string"/>
</s:sequence>
</s:extension>
</s:complexContent>
</s:complexType>
–
        <s:complexType name="BusinessObject" abstract="true">
–
        <s:sequence>
<s:element minOccurs="1" maxOccurs="1" name="Id" type="s:long"/>
</s:sequence>
</s:complexType>
–
        <s:complexType name="Trainee">
–
        <s:complexContent mixed="false">
–
        <s:extension base="tns:BusinessObject">
–
        <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="Job" type="s:string"/>
<s:element minOccurs="0" maxOccurs="1" name="Name" type="s:string"/>
</s:sequence>
</s:extension>
</s:complexContent>
</s:complexType>
–
        <s:complexType name="ArrayOfTrainee">
–
        <s:sequence>
```

```
<s:element minOccurs="0" maxOccurs="unbounded" name="Trainee" nillable="true" type="tns:Trainee"/>
</s:sequence>
</s:complexType>
–
        <s:element name="InsertObserverAssessmentPeriodResponse">
–
        <s:complexType>
–
        <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="InsertObserverAssessmentPeriodResult" type="tns:ObserverResult"/>
</s:sequence>
</s:complexType>
</s:element>
–
        <s:complexType name="ObserverResult">
–
        <s:sequence>
<s:element minOccurs="1" maxOccurs="1" name="StatusFlag" type="s:boolean"/>
<s:element minOccurs="0" maxOccurs="1" name="Value"/>
<s:element minOccurs="0" maxOccurs="1" name="Description" type="s:string"/>
<s:element minOccurs="0" maxOccurs="1" name="Message" type="s:string"/>
</s:sequence>
</s:complexType>
–
        <s:element name="InsertObserverAssessmentPeriodWithEndDate">
–
        <s:complexType>
–
        <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="moduleName" type="s:string"/>
<s:element minOccurs="1" maxOccurs="1" name="periodStartDate" type="s:dateTime"/>
<s:element minOccurs="1" maxOccurs="1" name="periodEndDate" type="s:dateTime"/>
<s:element minOccurs="0" maxOccurs="1" name="atts" type="tns:ArrayOfObserverAssessmentPeriodAttribute"/>
<s:element minOccurs="0" maxOccurs="1" name="trainees" type="tns:ArrayOfTrainee"/>
</s:sequence>
</s:complexType>
</s:element>
–
        <s:element name="InsertObserverAssessmentPeriodWithEndDateResponse">
–
        <s:complexType>
–
        <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="InsertObserverAssessmentPeriodWithEndDateResult"
type="tns:ObserverResult"/>
</s:sequence>
</s:complexType>
</s:element>
–
        <s:element name="UpdateObserverAssessmentPeriodEndDate">
–
        <s:complexType>
–
        <s:sequence>
<s:element minOccurs="1" maxOccurs="1" name="observerAssessmentPeriodId" type="s:long"/>
<s:element minOccurs="1" maxOccurs="1" name="periodEndDate" type="s:dateTime"/>
</s:sequence>
</s:complexType>
</s:element>
–
        <s:element name="UpdateObserverAssessmentPeriodEndDateResponse">
–
        <s:complexType>
–
        <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="UpdateObserverAssessmentPeriodEndDateResult" type="tns:ObserverResult"/>
</s:sequence>
</s:complexType>
```

```
</s:element>
–
        <s:element name="InsertPerformancePeriod">
–
        <s:complexType>
–
        <s:sequence>
<s:element minOccurs="1" maxOccurs="1" name="observerAssessmentPeriodId" type="s:long"/>
<s:element minOccurs="0" maxOccurs="1" name="performancePeriodName" type="s:string"/>
<s:element minOccurs="0" maxOccurs="1" name="atts" type="tns:ArrayOfPerformancePeriodAttribute"/>
</s:sequence>
</s:complexType>
</s:element>
–
        <s:complexType name="ArrayOfPerformancePeriodAttribute">
–
        <s:sequence>
<s:element minOccurs="0" maxOccurs="unbounded" name="PerformancePeriodAttribute" nillable="true"
type="tns:PerformancePeriodAttribute"/>
</s:sequence>
</s:complexType>
–
        <s:complexType name="PerformancePeriodAttribute">
–
        <s:complexContent mixed="false">
–
        <s:extension base="tns:BusinessObject">
–
        <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="AttributeName" type="s:string"/>
<s:element minOccurs="0" maxOccurs="1" name="AttributeValue" type="s:string"/>
</s:sequence>
</s:extension>
</s:complexContent>
</s:complexType>
–
        <s:element name="InsertPerformancePeriodResponse">
–
        <s:complexType>
–
        <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="InsertPerformancePeriodResult" type="tns:ObserverResult"/>
</s:sequence>
</s:complexType>
</s:element>
–
        <s:element name="InsertMeasurement">
–
        <s:complexType>
–
        <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="measurementQuestionText" type="s:string"/>
<s:element minOccurs="0" maxOccurs="1" name="measurementTypeName" type="s:string"/>
</s:sequence>
</s:complexType>
</s:element>
–
        <s:element name="InsertMeasurementResponse">
–
        <s:complexType>
–
        <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="InsertMeasurementResult" type="tns:ObserverResult"/>
</s:sequence>
</s:complexType>
</s:element>
–
        <s:element name="InsertMeasurementInstance">
```

```
–
        <s:complexType>
–
        <s:sequence>
<s:element minOccurs="1" maxOccurs="1" name="observerAssessmentPeriodId" type="s:long"/>
<s:element minOccurs="1" maxOccurs="1" name="measurementId" type="s:long"/>
<s:element minOccurs="1" maxOccurs="1" name="performancePeriodId" type="s:long"/>
<s:element minOccurs="1" maxOccurs="1" name="timestamp" type="s:dateTime"/>
<s:element minOccurs="0" maxOccurs="1" name="obs" type="tns:ArrayOfObservation"/>
</s:sequence>
</s:complexType>
</s:element>
–
        <s:complexType name="ArrayOfObservation">
–
        <s:sequence>
<s:element minOccurs="0" maxOccurs="unbounded" name="Observation" nillable="true" type="tns:Observation"/>
</s:sequence>
</s:complexType>
–
        <s:complexType name="Observation">
–
        <s:complexContent mixed="false">
–
        <s:extension base="tns:BusinessObject">
–
        <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="Value" type="s:string"/>
<s:element minOccurs="0" maxOccurs="1" name="DataType" type="s:string"/>
<s:element minOccurs="1" maxOccurs="1" name="IntVal" type="s:int"/>
<s:element minOccurs="1" maxOccurs="1" name="TraineeId" type="s:long"/>
</s:sequence>
</s:extension>
</s:complexContent>
</s:complexType>
–
        <s:element name="InsertMeasurementInstanceResponse">
–
        <s:complexType>
–
        <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="InsertMeasurementInstanceResult" type="tns:ObserverResult"/>
</s:sequence>
</s:complexType>
</s:element>
–
        <s:element name="UpdateMeasurementInstance">
–
        <s:complexType>
–
        <s:sequence>
<s:element minOccurs="1" maxOccurs="1" name="measurementInstanceId" type="s:long"/>
<s:element minOccurs="1" maxOccurs="1" name="timestamp" type="s:dateTime"/>
<s:element minOccurs="0" maxOccurs="1" name="obs" type="tns:ArrayOfObservation"/>
</s:sequence>
</s:complexType>
</s:element>
–
        <s:element name="UpdateMeasurementInstanceResponse">
–
        <s:complexType>
–
        <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="UpdateMeasurementInstanceResult" type="tns:ObserverResult"/>
</s:sequence>
</s:complexType>
</s:element>
–
```

```
        <s:element name="RemoveMeasurementInstance">
-
        <s:complexType>
-
        <s:sequence>
<s:element minOccurs="1" maxOccurs="1" name="measurementInstanceId" type="s:long"/>
</s:sequence>
</s:complexType>
</s:element>
-
        <s:element name="RemoveMeasurementInstanceResponse">
-
        <s:complexType>
-
        <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="RemoveMeasurementInstanceResult" type="tns:ObserverResult"/>
</s:sequence>
</s:complexType>
</s:element>
-
        <s:element name="InsertTrainee">
-
        <s:complexType>
-
        <s:sequence>
<s:element minOccurs="1" maxOccurs="1" name="observerAssessmentPeriodId" type="s:long"/>
<s:element minOccurs="0" maxOccurs="1" name="job" type="s:string"/>
<s:element minOccurs="0" maxOccurs="1" name="name" type="s:string"/>
</s:sequence>
</s:complexType>
</s:element>
-
        <s:element name="InsertTraineeResponse">
-
        <s:complexType>
-
        <s:sequence>
<s:element minOccurs="0" maxOccurs="1" name="InsertTraineeResult" type="tns:ObserverResult"/>
</s:sequence>
</s:complexType>
</s:element>
</s:schema>
</wsdl:types>
-
        <wsdl:message name="InsertObserverAssessmentPeriodSoapIn">
<wsdl:part name="parameters" element="tns:InsertObserverAssessmentPeriod"/>
</wsdl:message>
-
        <wsdl:message name="InsertObserverAssessmentPeriodSoapOut">
<wsdl:part name="parameters" element="tns:InsertObserverAssessmentPeriodResponse"/>
</wsdl:message>
-
        <wsdl:message name="InsertObserverAssessmentPeriodWithEndDateSoapIn">
<wsdl:part name="parameters" element="tns:InsertObserverAssessmentPeriodWithEndDate"/>
</wsdl:message>
-
        <wsdl:message name="InsertObserverAssessmentPeriodWithEndDateSoapOut">
<wsdl:part name="parameters" element="tns:InsertObserverAssessmentPeriodWithEndDateResponse"/>
</wsdl:message>
-
        <wsdl:message name="UpdateObserverAssessmentPeriodEndDateSoapIn">
<wsdl:part name="parameters" element="tns:UpdateObserverAssessmentPeriodEndDate"/>
</wsdl:message>
-
        <wsdl:message name="UpdateObserverAssessmentPeriodEndDateSoapOut">
<wsdl:part name="parameters" element="tns:UpdateObserverAssessmentPeriodEndDateResponse"/>
</wsdl:message>
-
```

```
        <wsdl:message name="InsertPerformancePeriodSoapIn">
<wsdl:part name="parameters" element="tns:InsertPerformancePeriod"/>
</wsdl:message>
-
        <wsdl:message name="InsertPerformancePeriodSoapOut">
<wsdl:part name="parameters" element="tns:InsertPerformancePeriodResponse"/>
</wsdl:message>
-
        <wsdl:message name="InsertMeasurementSoapIn">
<wsdl:part name="parameters" element="tns:InsertMeasurement"/>
</wsdl:message>
-
        <wsdl:message name="InsertMeasurementSoapOut">
<wsdl:part name="parameters" element="tns:InsertMeasurementResponse"/>
</wsdl:message>
-
        <wsdl:message name="InsertMeasurementInstanceSoapIn">
<wsdl:part name="parameters" element="tns:InsertMeasurementInstance"/>
</wsdl:message>
-
        <wsdl:message name="InsertMeasurementInstanceSoapOut">
<wsdl:part name="parameters" element="tns:InsertMeasurementInstanceResponse"/>
</wsdl:message>
-
        <wsdl:message name="UpdateMeasurementInstanceSoapIn">
<wsdl:part name="parameters" element="tns:UpdateMeasurementInstance"/>
</wsdl:message>
-
        <wsdl:message name="UpdateMeasurementInstanceSoapOut">
<wsdl:part name="parameters" element="tns:UpdateMeasurementInstanceResponse"/>
</wsdl:message>
-
        <wsdl:message name="RemoveMeasurementInstanceSoapIn">
<wsdl:part name="parameters" element="tns:RemoveMeasurementInstance"/>
</wsdl:message>
-
        <wsdl:message name="RemoveMeasurementInstanceSoapOut">
<wsdl:part name="parameters" element="tns:RemoveMeasurementInstanceResponse"/>
</wsdl:message>
-
        <wsdl:message name="InsertTraineeSoapIn">
<wsdl:part name="parameters" element="tns:InsertTrainee"/>
</wsdl:message>
-
        <wsdl:message name="InsertTraineeSoapOut">
<wsdl:part name="parameters" element="tns:InsertTraineeResponse"/>
</wsdl:message>
-
        <wsdl:portType name="ServiceObserverMeasureSoap">
-
        <wsdl:operation name="InsertObserverAssessmentPeriod">
-
        <wsdl:documentation>
Create a new <b>observer assessment period and attributes</b> for a specific starting date.
</wsdl:documentation>
<wsdl:input message="tns:InsertObserverAssessmentPeriodSoapIn"/>
<wsdl:output message="tns:InsertObserverAssessmentPeriodSoapOut"/>
</wsdl:operation>
-
        <wsdl:operation name="InsertObserverAssessmentPeriodWithEndDate">
-
        <wsdl:documentation>
Create a new <b>observer assessment period and attributes</b> for a specific date range.
</wsdl:documentation>
<wsdl:input message="tns:InsertObserverAssessmentPeriodWithEndDateSoapIn"/>
<wsdl:output message="tns:InsertObserverAssessmentPeriodWithEndDateSoapOut"/>
</wsdl:operation>
-
```

```
            <wsdl:operation name="UpdateObserverAssessmentPeriodEndDate">
–
            <wsdl:documentation>
Update an existing <b>observer assessment period end date</b>.
</wsdl:documentation>
<wsdl:input message="tns:UpdateObserverAssessmentPeriodEndDateSoapIn"/>
<wsdl:output message="tns:UpdateObserverAssessmentPeriodEndDateSoapOut"/>
</wsdl:operation>
–
            <wsdl:operation name="InsertPerformancePeriod">
–
            <wsdl:documentation>
Create a new <b>performance period and attributes</b> for an observer assessment.
</wsdl:documentation>
<wsdl:input message="tns:InsertPerformancePeriodSoapIn"/>
<wsdl:output message="tns:InsertPerformancePeriodSoapOut"/>
</wsdl:operation>
–
            <wsdl:operation name="InsertMeasurement">
–
            <wsdl:documentation>
Create a new <b>measurement</b> for a specific question and measurement type.
</wsdl:documentation>
<wsdl:input message="tns:InsertMeasurementSoapIn"/>
<wsdl:output message="tns:InsertMeasurementSoapOut"/>
</wsdl:operation>
–
            <wsdl:operation name="InsertMeasurementInstance">
–
            <wsdl:documentation>
Create a new <b>measurement instance and observations</b> for an observer assessment period.
</wsdl:documentation>
<wsdl:input message="tns:InsertMeasurementInstanceSoapIn"/>
<wsdl:output message="tns:InsertMeasurementInstanceSoapOut"/>
</wsdl:operation>
–
            <wsdl:operation name="UpdateMeasurementInstance">
–
            <wsdl:documentation>
Update an existing <b>measurement instance and observations</b>.
</wsdl:documentation>
<wsdl:input message="tns:UpdateMeasurementInstanceSoapIn"/>
<wsdl:output message="tns:UpdateMeasurementInstanceSoapOut"/>
</wsdl:operation>
–
            <wsdl:operation name="RemoveMeasurementInstance">
–
            <wsdl:documentation>
Remove an existing <b>measurement instance and observations</b>.
</wsdl:documentation>
<wsdl:input message="tns:RemoveMeasurementInstanceSoapIn"/>
<wsdl:output message="tns:RemoveMeasurementInstanceSoapOut"/>
</wsdl:operation>
–
            <wsdl:operation name="InsertTrainee">
–
            <wsdl:documentation>
Create a new <b>trainee</b> for a specific observer assessment period.
</wsdl:documentation>
<wsdl:input message="tns:InsertTraineeSoapIn"/>
<wsdl:output message="tns:InsertTraineeSoapOut"/>
</wsdl:operation>
</wsdl:portType>
–
            <wsdl:binding name="ServiceObserverMeasureSoap" type="tns:ServiceObserverMeasureSoap">
<soap:binding transport="http://schemas.xmlsoap.org/soap/http"/>
–
            <wsdl:operation name="InsertObserverAssessmentPeriod">
```

```
<soap:operation soapAction="http://PMOEngine.aptima.com/InsertObserverAssessmentPeriod" style="document"/>
–
        <wsdl:input>
<soap:body use="literal"/>
</wsdl:input>
–
        <wsdl:output>
<soap:body use="literal"/>
</wsdl:output>
</wsdl:operation>
–
        <wsdl:operation name="InsertObserverAssessmentPeriodWithEndDate">
<soap:operation soapAction="http://PMOEngine.aptima.com/InsertObserverAssessmentPeriodWithEndDate" style="document"/>
–
        <wsdl:input>
<soap:body use="literal"/>
</wsdl:input>
–
        <wsdl:output>
<soap:body use="literal"/>
</wsdl:output>
</wsdl:operation>
–
        <wsdl:operation name="UpdateObserverAssessmentPeriodEndDate">
<soap:operation soapAction="http://PMOEngine.aptima.com/UpdateObserverAssessmentPeriodEndDate" style="document"/>
–
        <wsdl:input>
<soap:body use="literal"/>
</wsdl:input>
–
        <wsdl:output>
<soap:body use="literal"/>
</wsdl:output>
</wsdl:operation>
–
        <wsdl:operation name="InsertPerformancePeriod">
<soap:operation soapAction="http://PMOEngine.aptima.com/InsertPerformancePeriod" style="document"/>
–
        <wsdl:input>
<soap:body use="literal"/>
</wsdl:input>
–
        <wsdl:output>
<soap:body use="literal"/>
</wsdl:output>
</wsdl:operation>
–
        <wsdl:operation name="InsertMeasurement">
<soap:operation soapAction="http://PMOEngine.aptima.com/InsertMeasurement" style="document"/>
–
        <wsdl:input>
<soap:body use="literal"/>
</wsdl:input>
–
        <wsdl:output>
<soap:body use="literal"/>
</wsdl:output>
</wsdl:operation>
–
        <wsdl:operation name="InsertMeasurementInstance">
<soap:operation soapAction="http://PMOEngine.aptima.com/InsertMeasurementInstance" style="document"/>
–
        <wsdl:input>
<soap:body use="literal"/>
</wsdl:input>
–
        <wsdl:output>
<soap:body use="literal"/>
```

```
</wsdl:output>
</wsdl:operation>
–
        <wsdl:operation name="UpdateMeasurementInstance">
<soap:operation soapAction="http://PMOEngine.aptima.com/UpdateMeasurementInstance" style="document"/>
–
        <wsdl:input>
<soap:body use="literal"/>
</wsdl:input>
–
        <wsdl:output>
<soap:body use="literal"/>
</wsdl:output>
</wsdl:operation>
–
        <wsdl:operation name="RemoveMeasurementInstance">
<soap:operation soapAction="http://PMOEngine.aptima.com/RemoveMeasurementInstance" style="document"/>
–
        <wsdl:input>
<soap:body use="literal"/>
</wsdl:input>
–
        <wsdl:output>
<soap:body use="literal"/>
</wsdl:output>
</wsdl:operation>
–
        <wsdl:operation name="InsertTrainee">
<soap:operation soapAction="http://PMOEngine.aptima.com/InsertTrainee" style="document"/>
–
        <wsdl:input>
<soap:body use="literal"/>
</wsdl:input>
–
        <wsdl:output>
<soap:body use="literal"/>
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
–
        <wsdl:binding name="ServiceObserverMeasureSoap12" type="tns:ServiceObserverMeasureSoap">
<soap12:binding transport="http://schemas.xmlsoap.org/soap/http"/>
–
        <wsdl:operation name="InsertObserverAssessmentPeriod">
<soap12:operation soapAction="http://PMOEngine.aptima.com/InsertObserverAssessmentPeriod" style="document"/>
–
        <wsdl:input>
<soap12:body use="literal"/>
</wsdl:input>
–
        <wsdl:output>
<soap12:body use="literal"/>
</wsdl:output>
</wsdl:operation>
–
        <wsdl:operation name="InsertObserverAssessmentPeriodWithEndDate">
<soap12:operation soapAction="http://PMOEngine.aptima.com/InsertObserverAssessmentPeriodWithEndDate" style="document"/>
–
        <wsdl:input>
<soap12:body use="literal"/>
</wsdl:input>
–
        <wsdl:output>
<soap12:body use="literal"/>
</wsdl:output>
</wsdl:operation>
–
        <wsdl:operation name="UpdateObserverAssessmentPeriodEndDate">
```

<soap12:operation soapAction="http://PMOEngine.aptima.com/UpdateObserverAssessmentPeriodEndDate" style="document"/>
–
<wsdl:input>
<soap12:body use="literal"/>
</wsdl:input>
–
<wsdl:output>
<soap12:body use="literal"/>
</wsdl:output>
</wsdl:operation>
–
<wsdl:operation name="InsertPerformancePeriod">
<soap12:operation soapAction="http://PMOEngine.aptima.com/InsertPerformancePeriod" style="document"/>
–
<wsdl:input>
<soap12:body use="literal"/>
</wsdl:input>
–
<wsdl:output>
<soap12:body use="literal"/>
</wsdl:output>
</wsdl:operation>
–
<wsdl:operation name="InsertMeasurement">
<soap12:operation soapAction="http://PMOEngine.aptima.com/InsertMeasurement" style="document"/>
–
<wsdl:input>
<soap12:body use="literal"/>
</wsdl:input>
–
<wsdl:output>
<soap12:body use="literal"/>
</wsdl:output>
</wsdl:operation>
–
<wsdl:operation name="InsertMeasurementInstance">
<soap12:operation soapAction="http://PMOEngine.aptima.com/InsertMeasurementInstance" style="document"/>
–
<wsdl:input>
<soap12:body use="literal"/>
</wsdl:input>
–
<wsdl:output>
<soap12:body use="literal"/>
</wsdl:output>
</wsdl:operation>
–
<wsdl:operation name="UpdateMeasurementInstance">
<soap12:operation soapAction="http://PMOEngine.aptima.com/UpdateMeasurementInstance" style="document"/>
–
<wsdl:input>
<soap12:body use="literal"/>
</wsdl:input>
–
<wsdl:output>
<soap12:body use="literal"/>
</wsdl:output>
</wsdl:operation>
–
<wsdl:operation name="RemoveMeasurementInstance">
<soap12:operation soapAction="http://PMOEngine.aptima.com/RemoveMeasurementInstance" style="document"/>
–
<wsdl:input>
<soap12:body use="literal"/>
</wsdl:input>
–
<wsdl:output>
<soap12:body use="literal"/>

```
</wsdl:output>
</wsdl:operation>
-
        <wsdl:operation name="InsertTrainee">
<soap12:operation soapAction="http://PMOEngine.aptima.com/InsertTrainee" style="document"/>
-
        <wsdl:input>
<soap12:body use="literal"/>
</wsdl:input>
-
        <wsdl:output>
<soap12:body use="literal"/>
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
-
        <wsdl:service name="ServiceObserverMeasure">
<wsdl:documentation><b>PMO Engine - Observer Measure Service</b></wsdl:documentation>
-
        <wsdl:port name="ServiceObserverMeasureSoap" binding="tns:ServiceObserverMeasureSoap">
<soap:address location="http://localhost/ObserverMeasure/ServiceObserverMeasure.asmx"/>
</wsdl:port>
-
        <wsdl:port name="ServiceObserverMeasureSoap12" binding="tns:ServiceObserverMeasureSoap12">
<soap12:address location="http://localhost/ObserverMeasure/ServiceObserverMeasure.asmx"/>
</wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

Appendix II

Installation Manual

This document details the steps necessary to configure a computer to run the Observer Assessment Web Service application.

These instructions assume that the user has administrative privileges on the computer that the software will be installed. The software has been designed to run on a Windows operating system. If you have any questions about the installation procedures, please contact Jeanine Ayers at Aptima, Inc (phone: 781-496-2489, email: jayers@aptima.com).

Operating System Requirements

A Windows NT based operating system (NT, 2000, XP, 2003) is required to run the Observer Assessment Web Service software and database.

Software Requirements

The Observer Assessment Web Service software requires a MySQL database. MySQL is an open-source database server that is free and available over the internet. A version of the installation software for MySQL has been included on the distribution CD in the following directory structure:

Z:\Required Software

The Observer Assessment Web Service application software can be found on the distribution CD in the following directory structure:

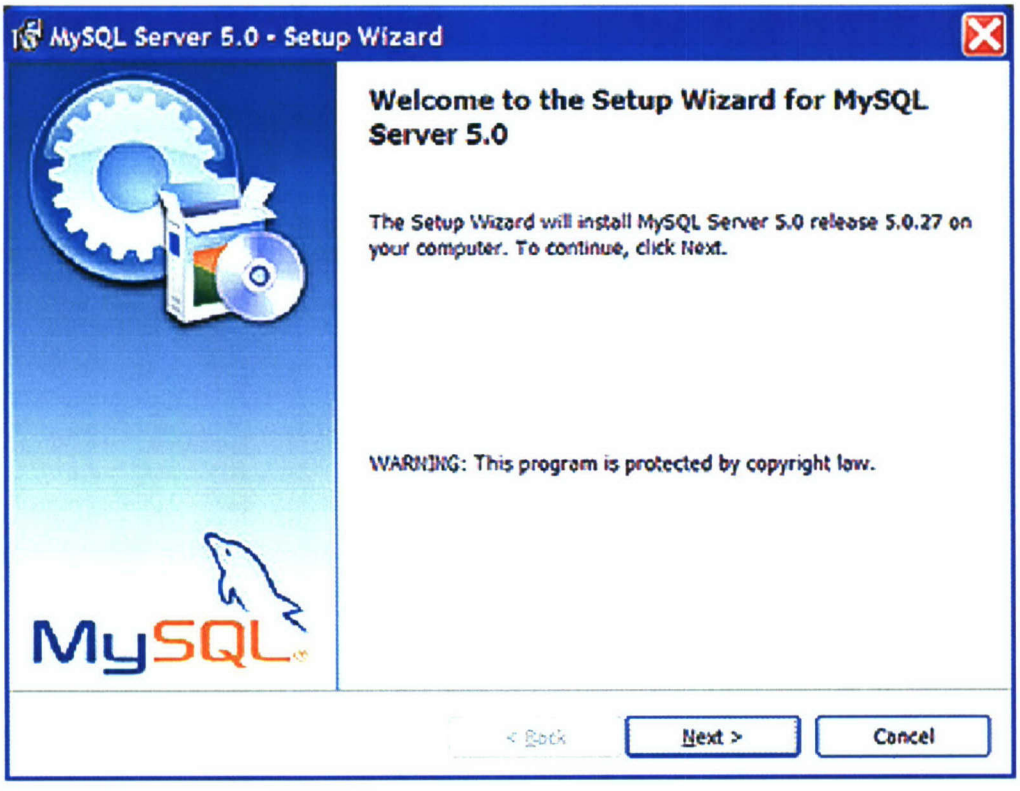
Z:\ ObserverMeasureWebSetup.msi

This installation document describes in detail the three steps that a user must execute to get the Observer Assessment Web Service software running on a computer.

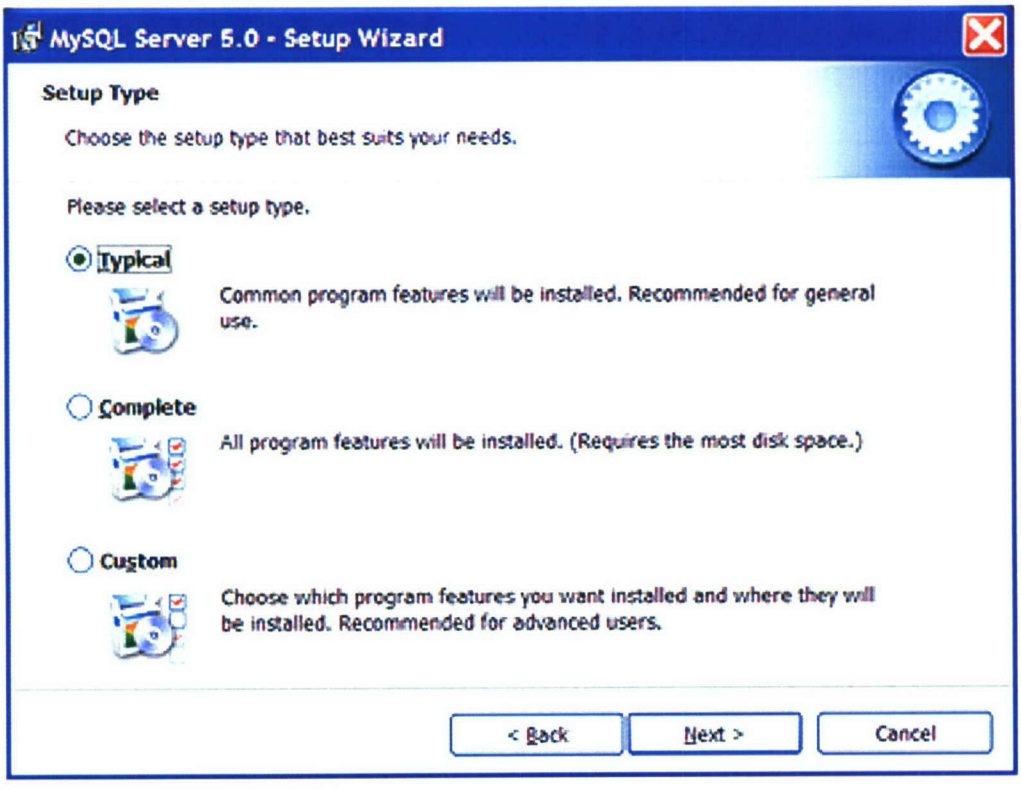
I. Install MySQL Server 5.0

Unzip the contents of **mysql-5.0.27-win32.zip** file in the Required Software folder to a temporary folder.

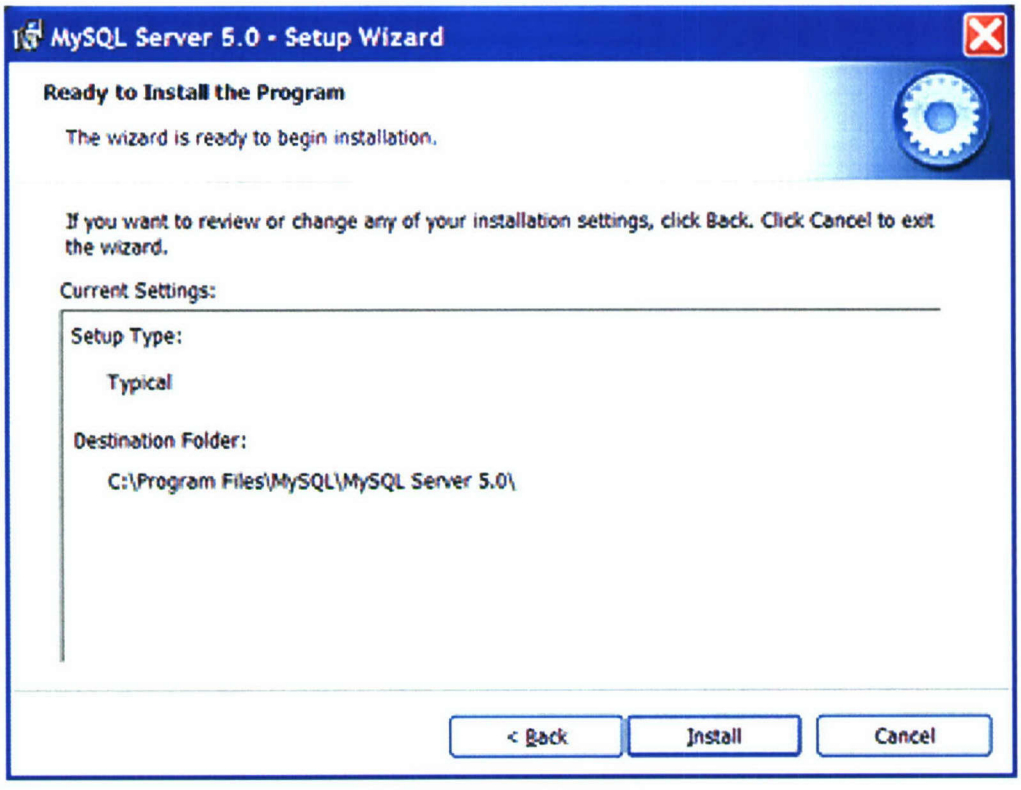
In the temporary folder, double click the **Setup.exe** file.



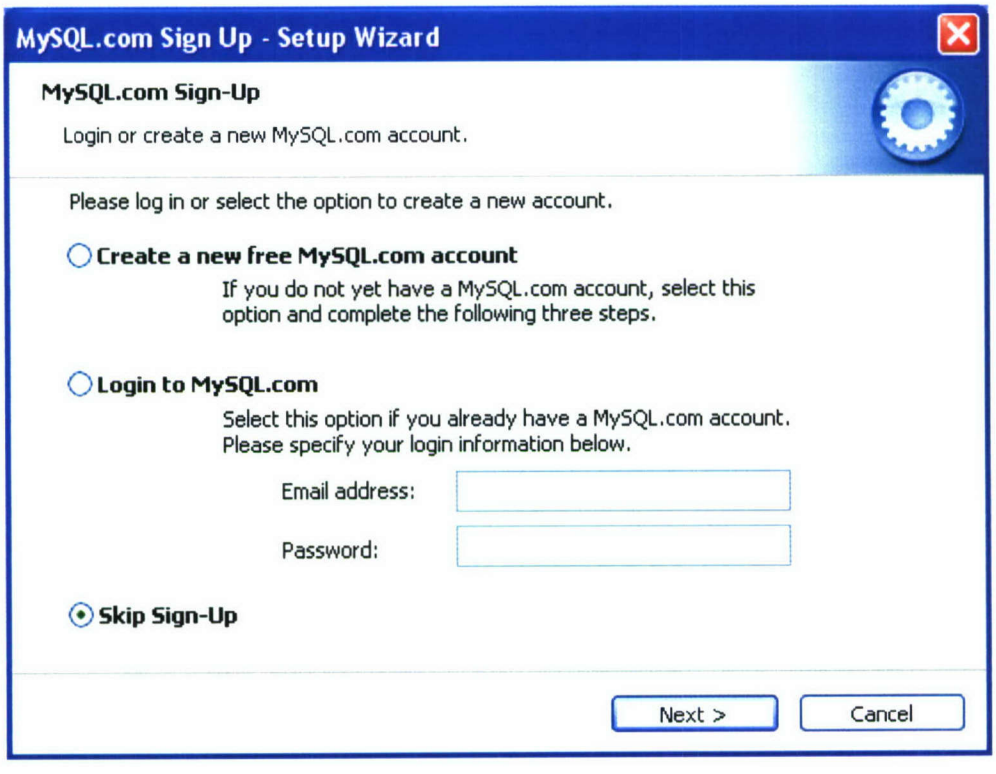
Press the "**Next**" button to continue.



Select the "**Typical**" radio button option and press the "**Next**" button to continue.

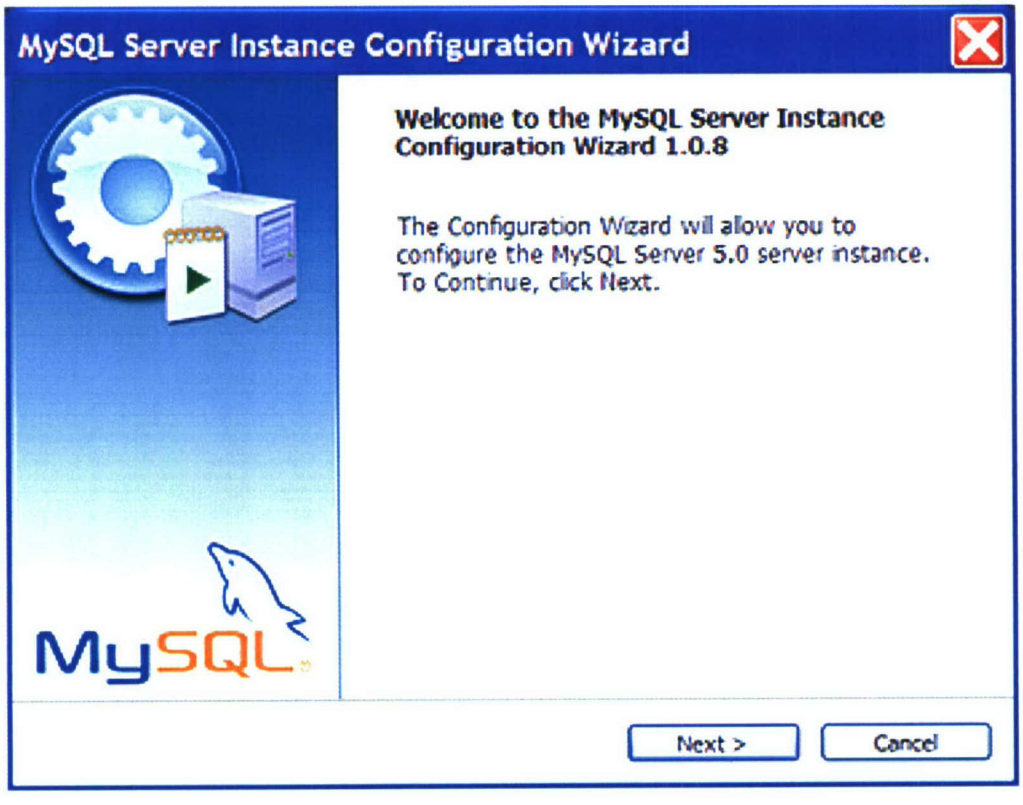


Press the "**Install**" button to install MySQL Server 5.0.



Select the "**Skip Sign-Up**" radio button option and press the "**Next**" button to continue.

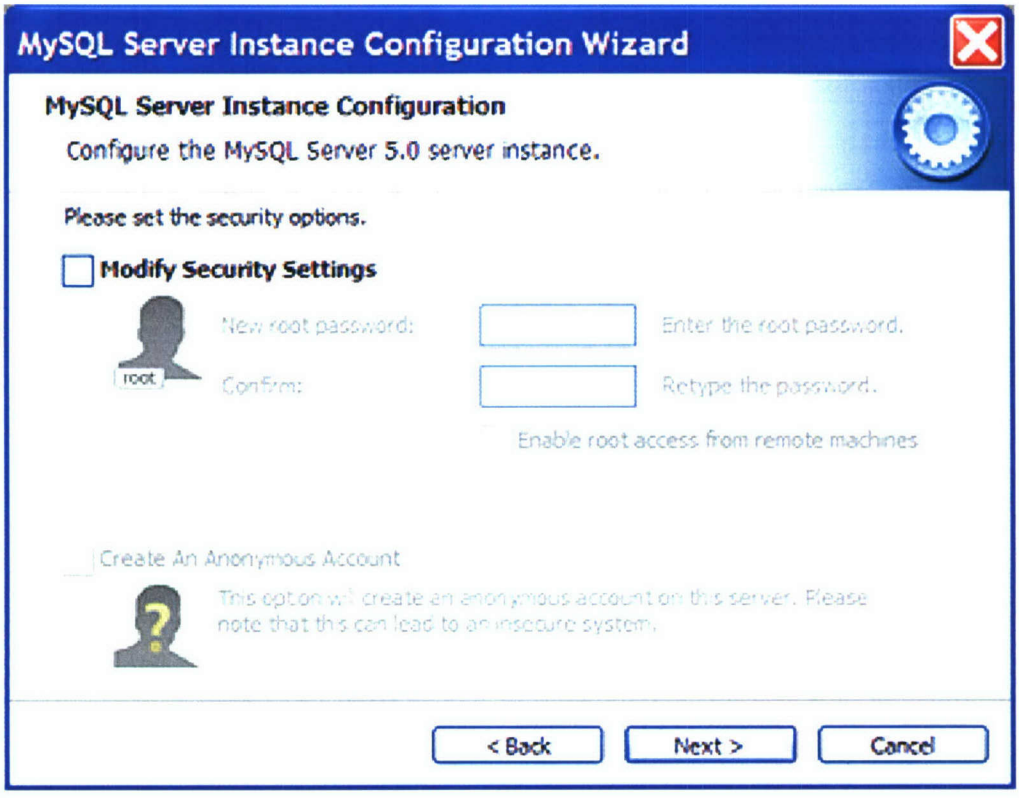
Check "**Configure the MySQL Server now**" and press the "**Finish"** button.



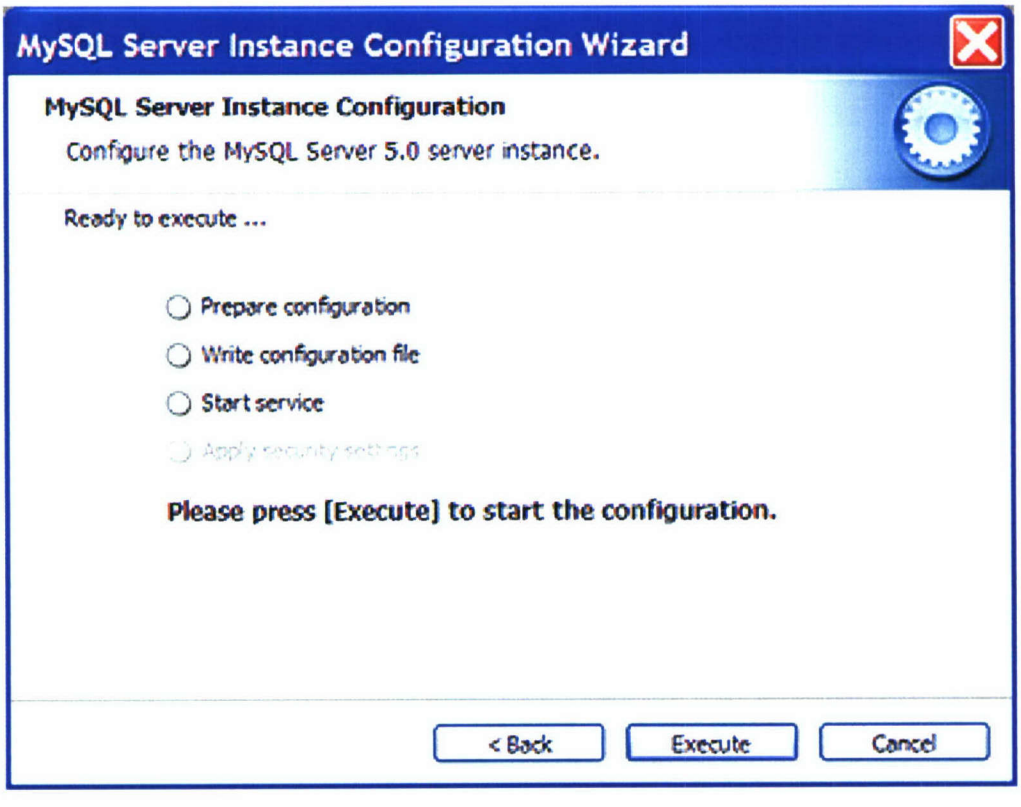
Press the "**Next**" button to continue.

Select the "**Standard Configuration**" radio button option and press the "**Next**" button to continue.

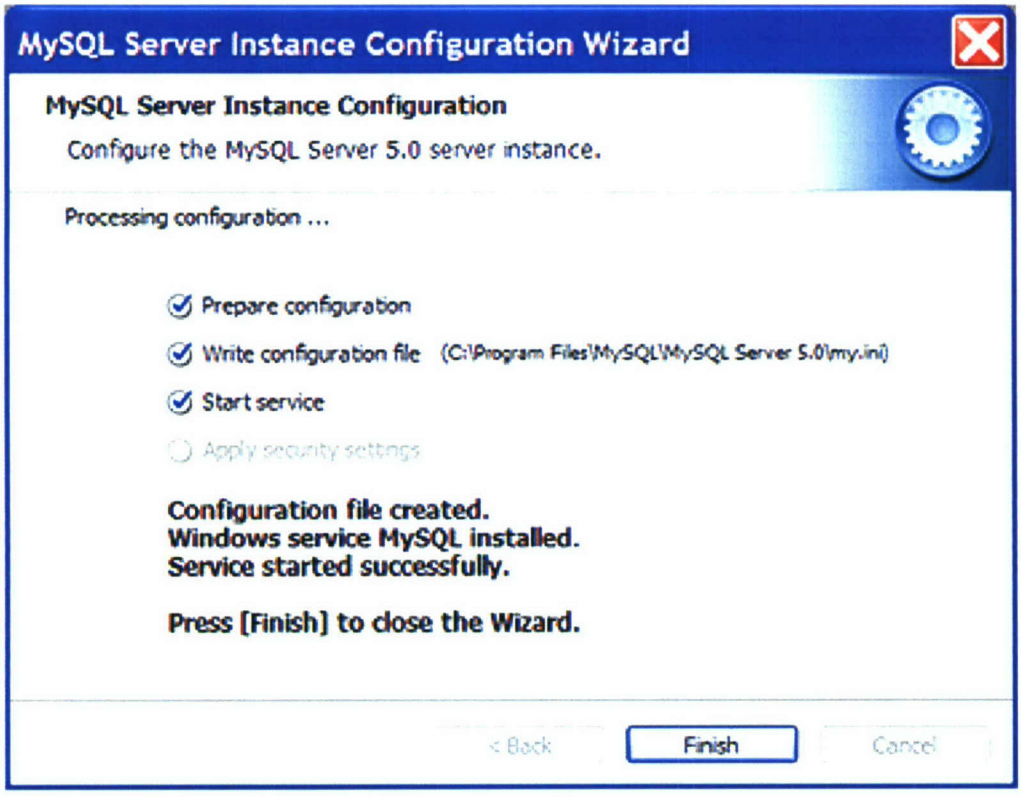
Check "**Install As Windows Service**", "**Launch the MySQL Server automatically**", and "**Include Bin Directory in Windows PATH**". Press the "**Next**" button to continue.



Uncheck "**Modify Security Settings**". Press the "**Next**" button to continue.



Press the "**Execute**" button to configure the MySQL Server 5.0 server instance.

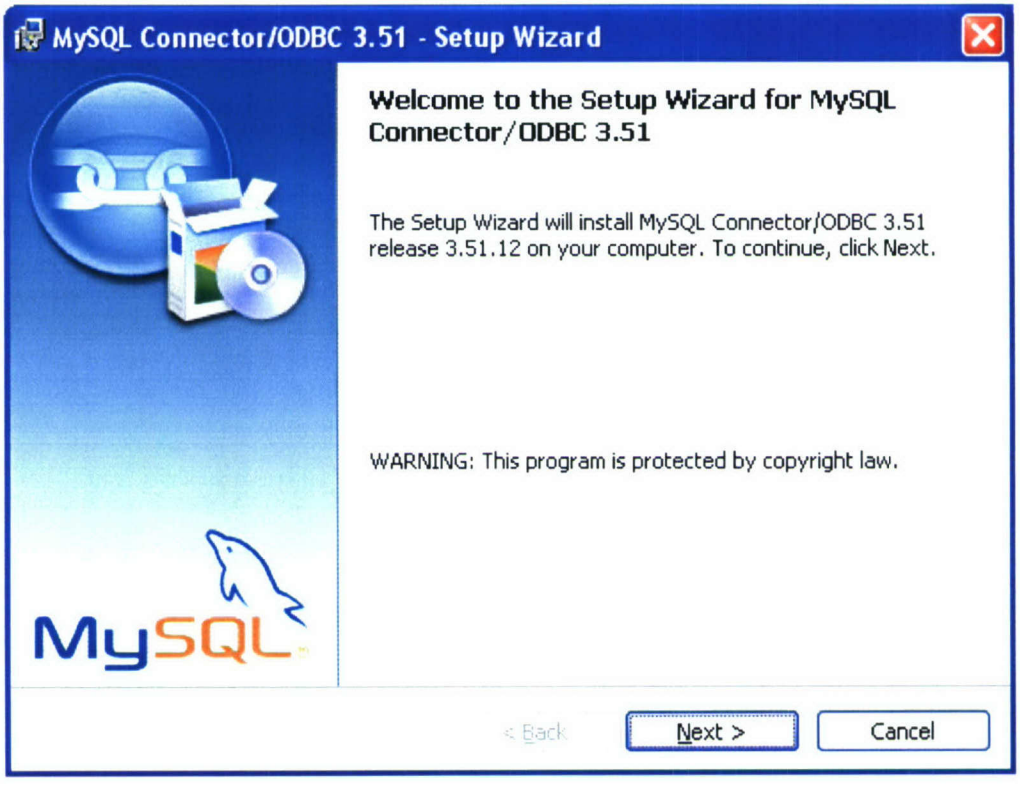


Press the "**Finish**" button to close the wizard.

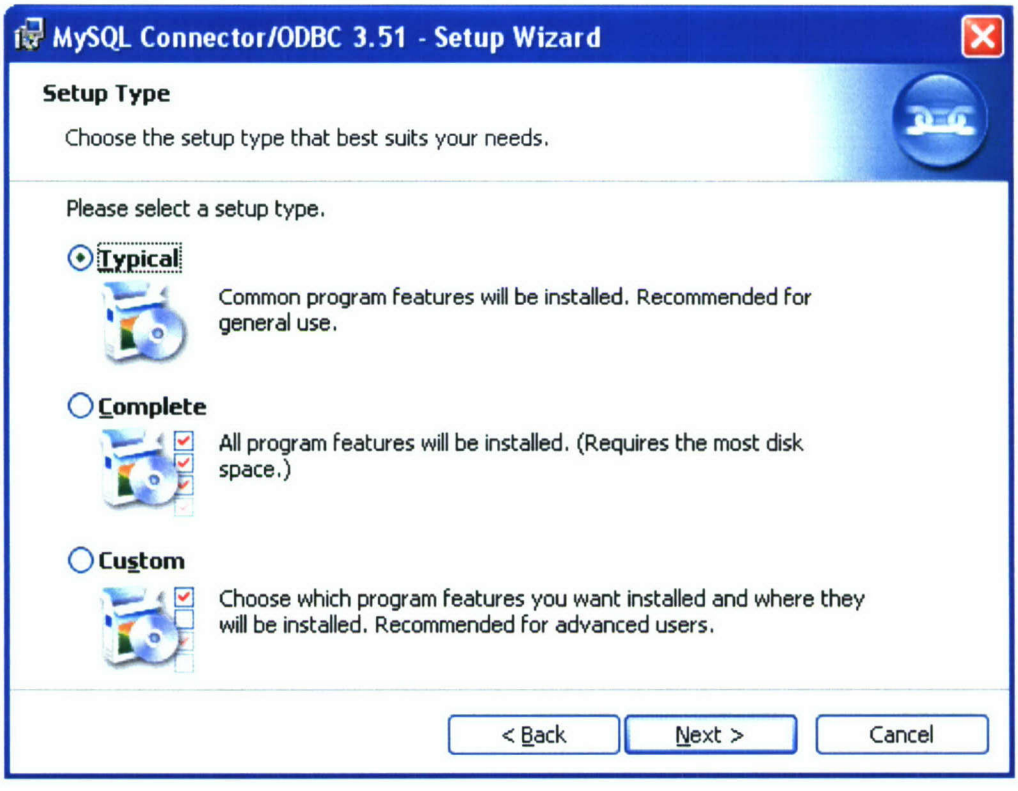
II. Install MySQL Connector/ODBC 3.51

Unzip the contents of **mysql-connector-odbc-3.51.12-win32.zip** file in the Required Software folder to a temporary folder.

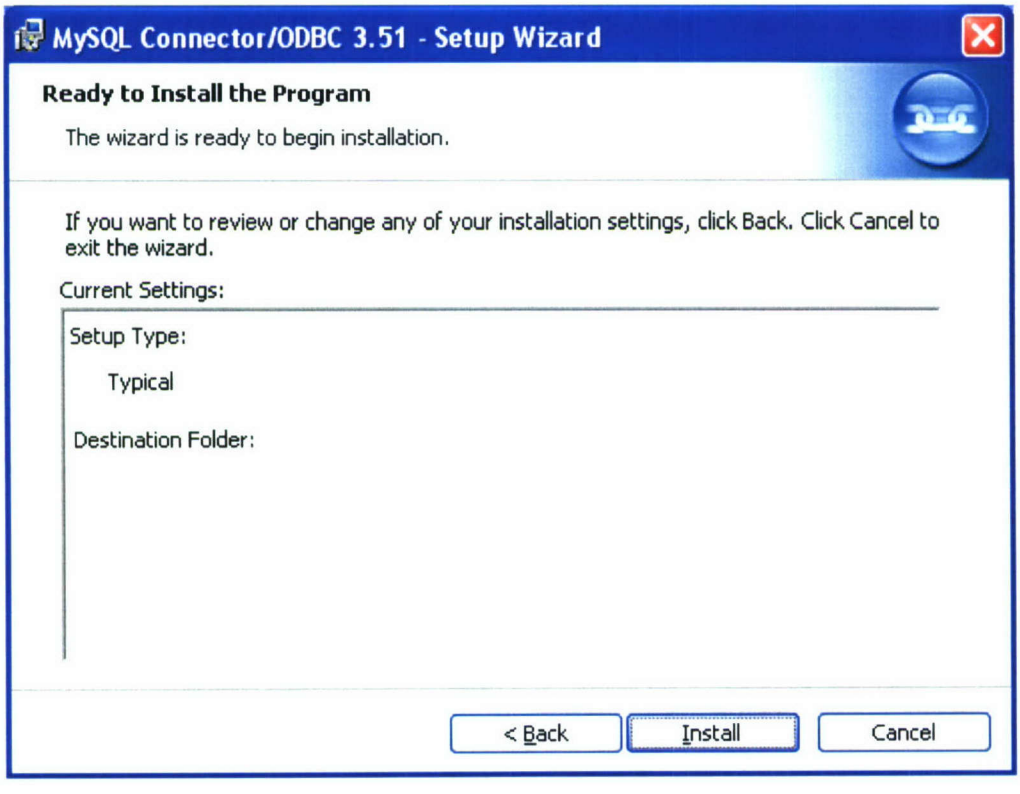
In the temporary folder, double click the **Setup.exe** file.



Press the "**Next**" button to continue.



Check the "**Typical**" radio button option and press the "**Next**" button to continue.

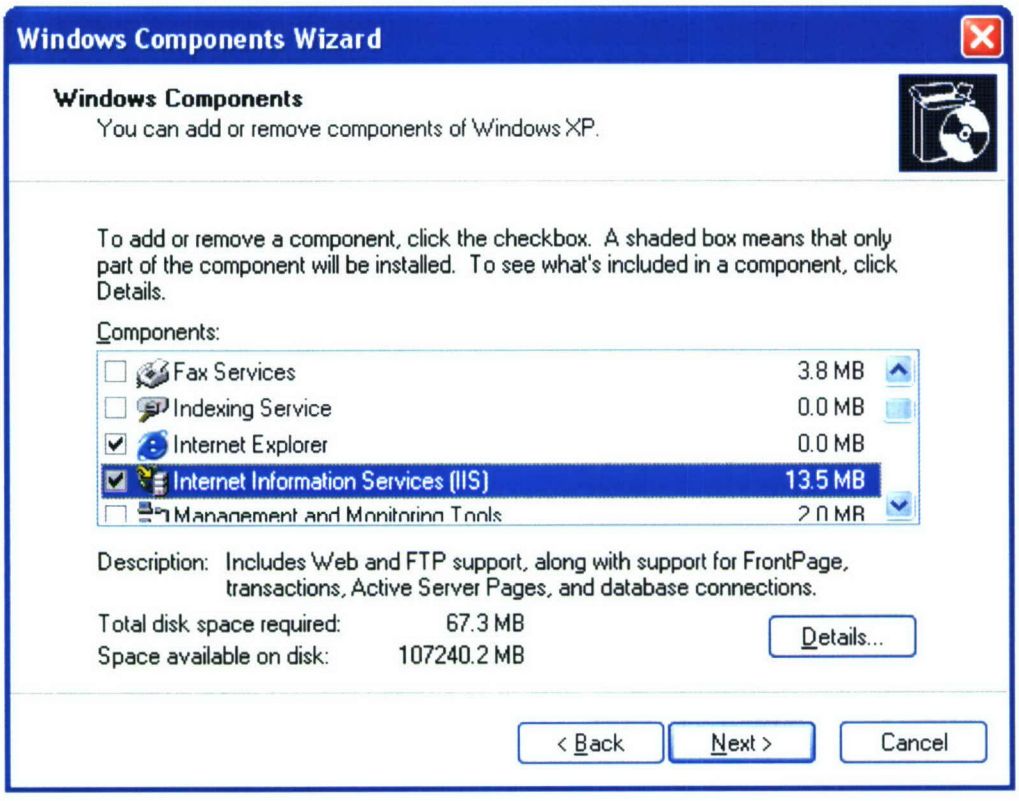


Press the "**Install**" button to install MySQL Connector/ODBC 3.51.

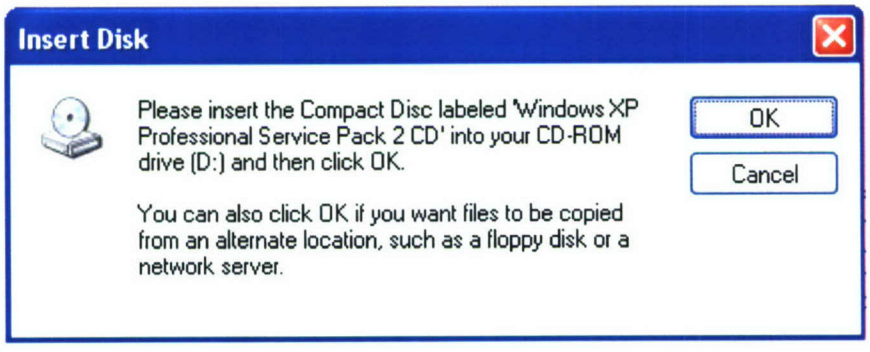
Press the "**Finish**" button to exit the wizard.

III. IIS 5.1

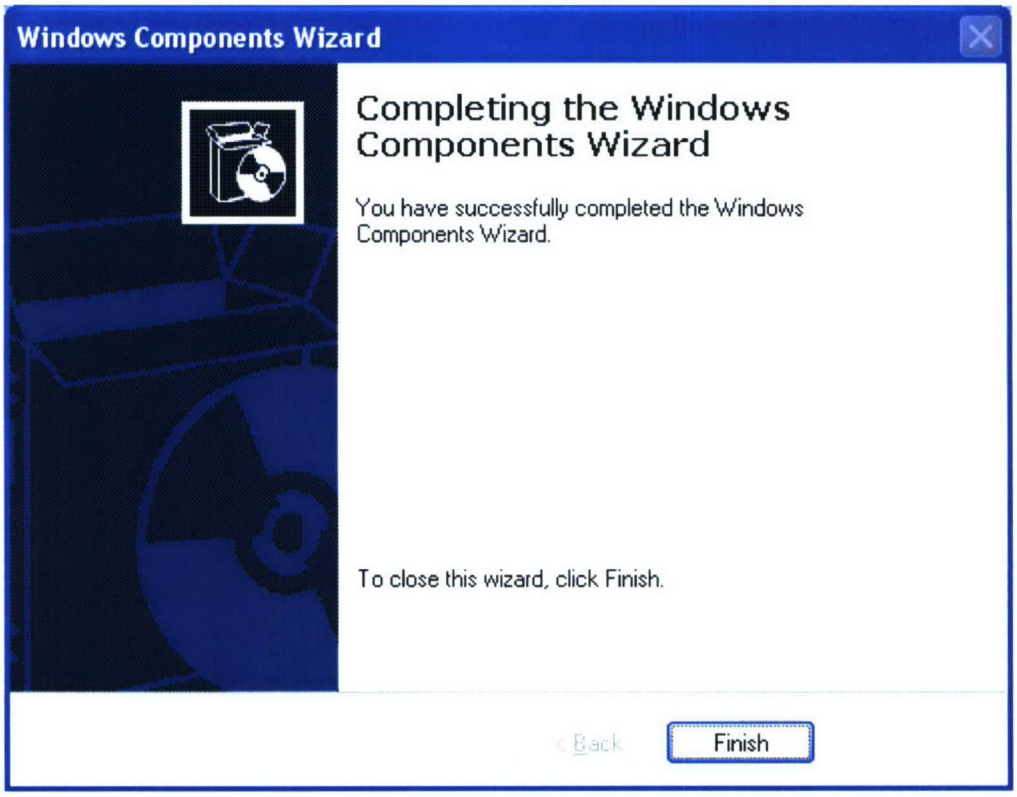
This must be installed prior to installing the Microsoft .NET Framework 2.0. Press the "**Start**" Taskbar button. Select the "**Control Panel**" menu item. Click "**Add or Remove Programs.**" Click **"Add/Remove Windows Components".**



Check "**Internet Information Services (IIS)**" and press the "**Next**" button to continue.



Insert the Compact Disc labeled "Windows **XP Professional Service Pack 2 CD**" into your CD-ROM drive and the press the "**OK" button**".

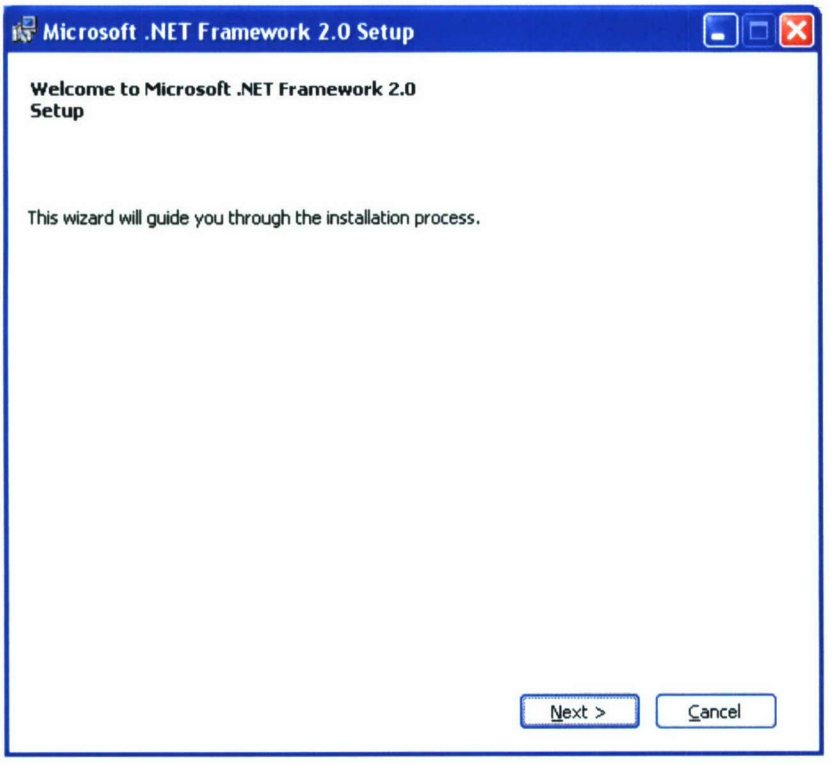


Press the "**Finish**" button to exit the wizard.

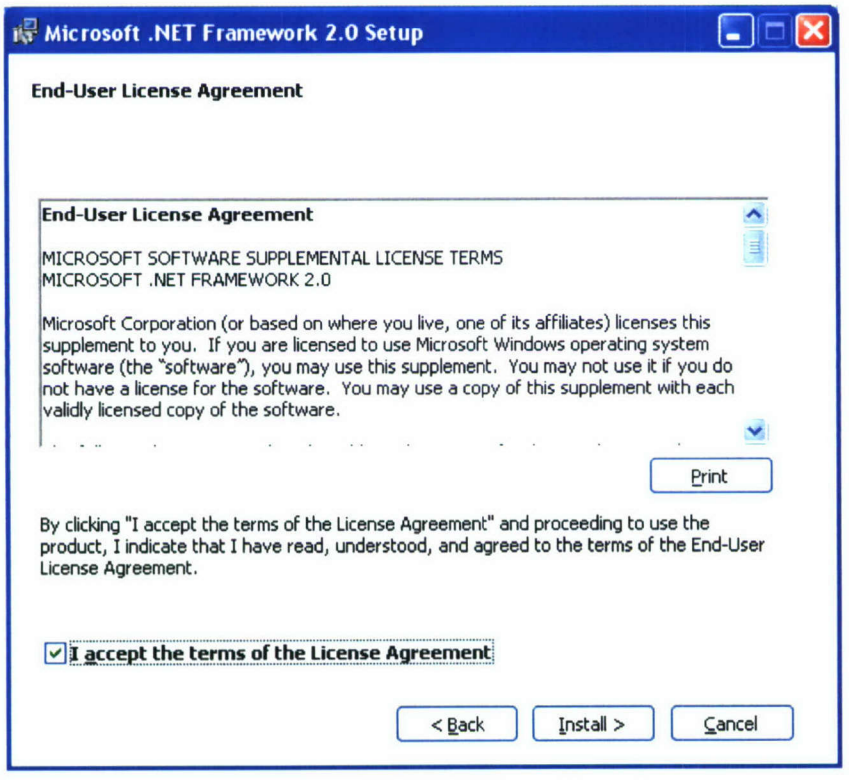
Press the "**Start**" Taskbar button. Select the "**Control Panel**" menu item. Click each of the following in the same exact order: "**Performance and Maintenance**", "**Administrative Tools**", and "**Services**". Right Click on the "**IIS Admin**" service and select the "**Stop**" menu item.

IV. Install Microsoft .NET Framework 2.0

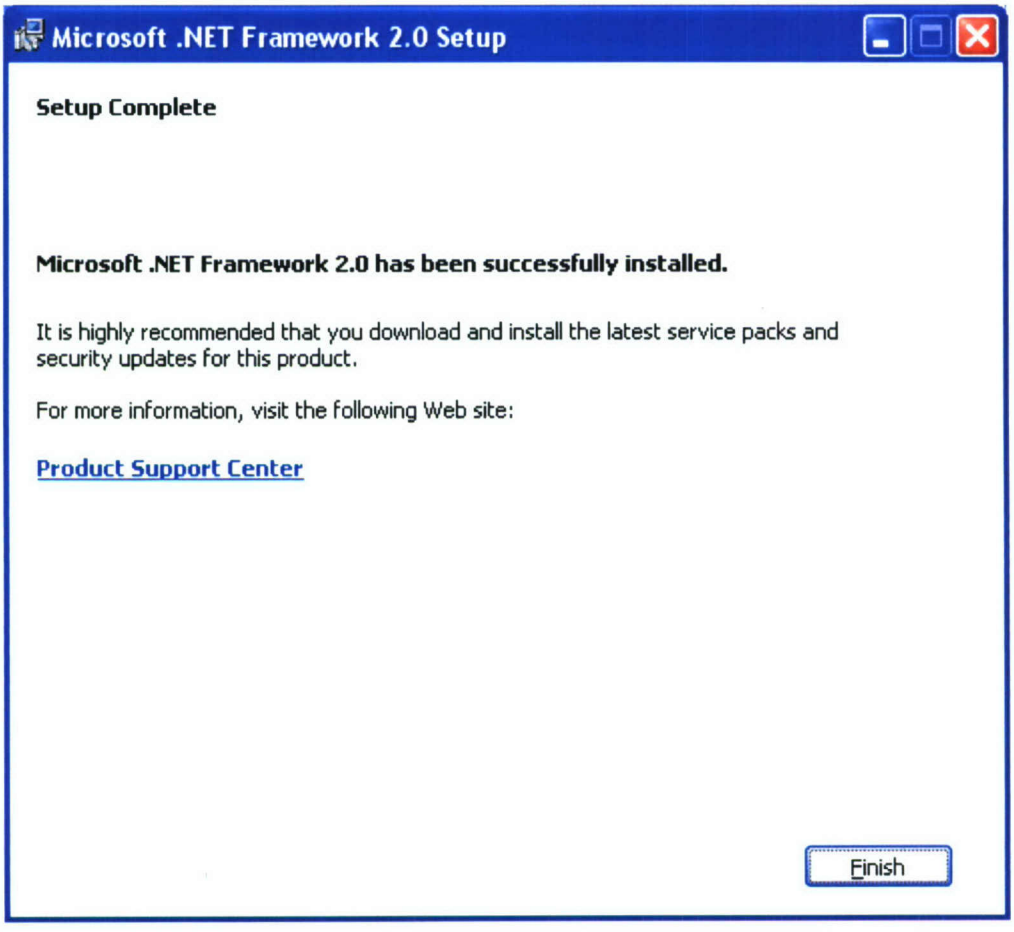
Double click the **dotnetfx2.0-x86.exe** file in the Aptima Software folder.



Press the "**Next**" button to continue.



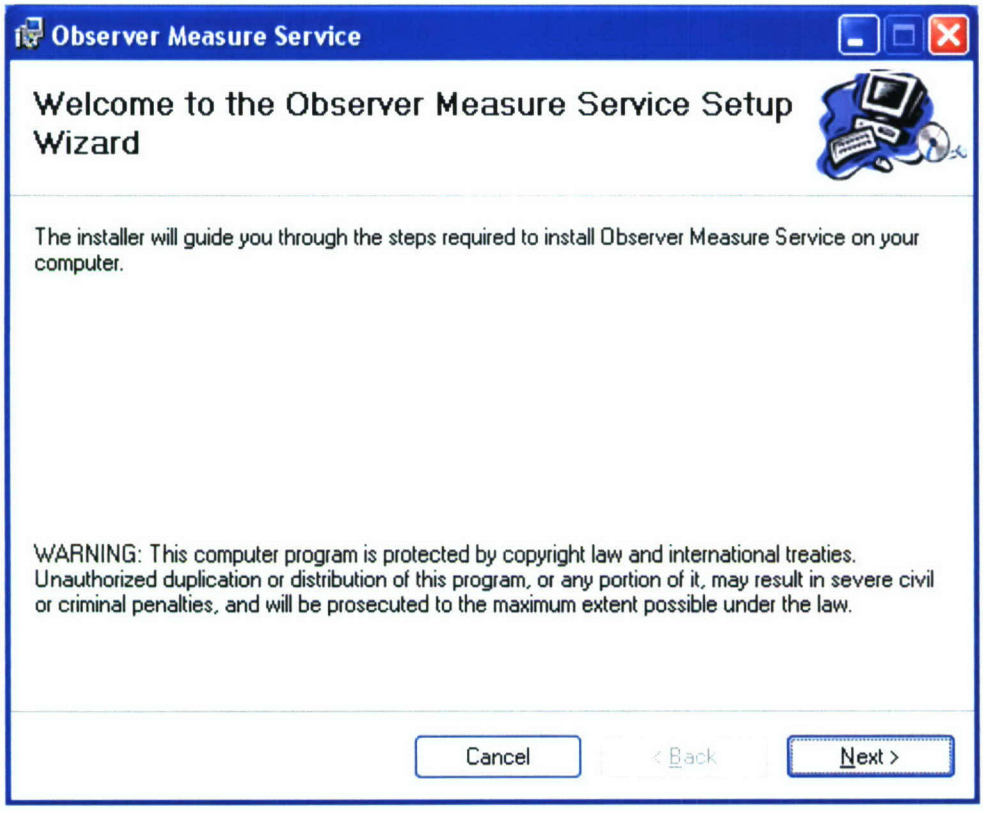
Read the end-user license agreement. Check the "**I accept the terms of the License Agreement.**" Press the "**Install**" button to continue.



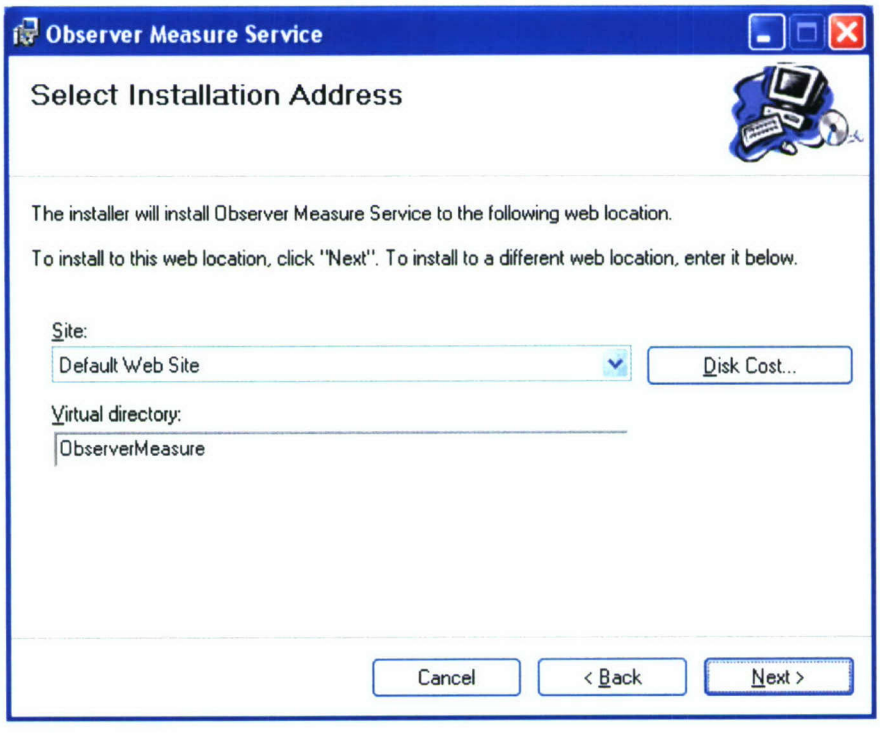
Press the "**Finish**" button to exit the wizard.

V. Install the Observer Measure Service

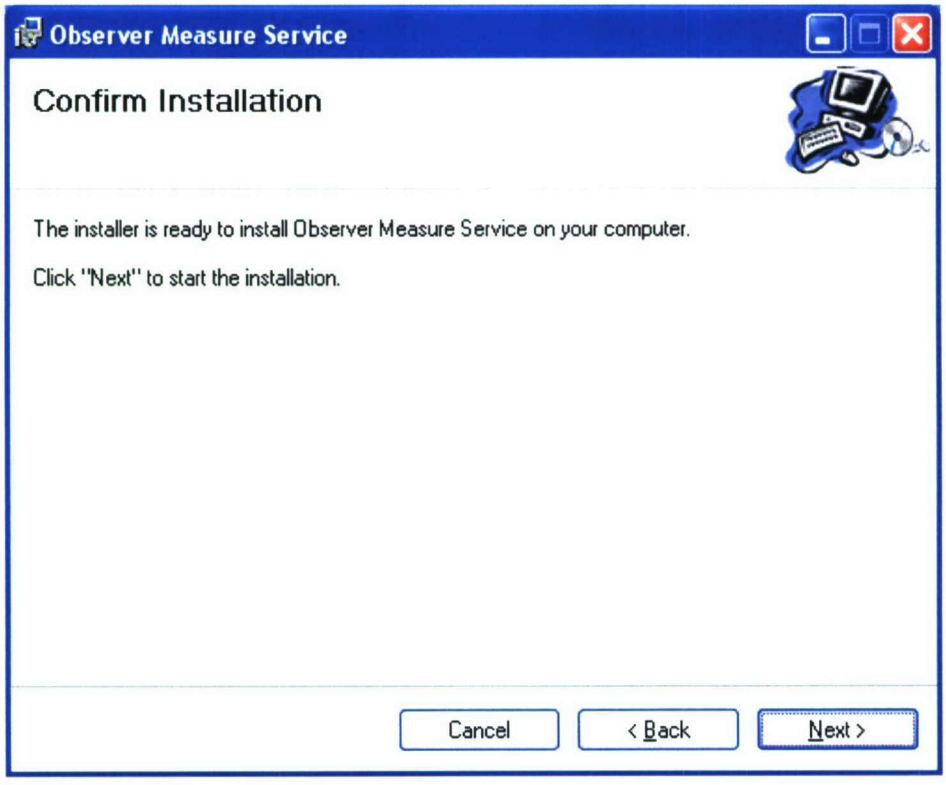
Double click the **ObserverMeasureWebSetup.msi** file in the Aptima Software\SOM folder.



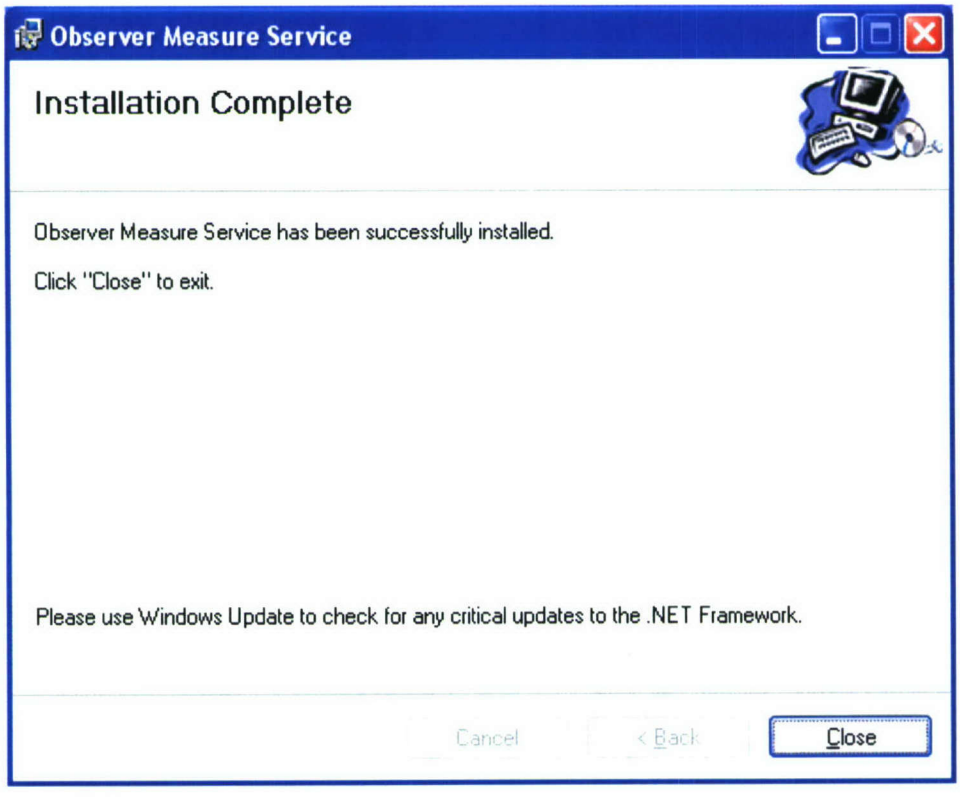
Press the "**Next**" button to continue.



Do not change the default for Site and Virtual Directory. The default for Site should equal "**Default Web Site**" and the default for Virtual Directory should equal "**ObserverMeasure**". Press the "**Next**" button to continue.



Press the "**Next**" button to continue.



Press the "**Close**" button to exit the wizard.